Reflowing Digital Ink Annotations

David Bargeron, Tomer Moscovich[1]
Microsoft Research
One Microsoft Way
Redmond, WA, 98052 USA
+1 425 703 7526
davemb@microsoft.com, tm@cs.brown.edu

ABSTRACT
Annotating paper documents with a pen is a familiar and indispensable activity across a wide variety of work and educational settings. Recent developments in pen-based computing promise to bring this experience to digital documents. However, digital documents are more flexible than their paper counterparts. When a digital document is edited, or displayed on different devices, its layout adapts to the new situation. Freeform digital ink annotations made on such a document must likewise adapt, or "reflow." But their unconstrained nature yields only vague guidelines for how these annotations should be transformed. Few systems have considered this issue, and still fewer have addressed it from a user's point of view. This paper reports the results of a study of user expectations for reflowing digital ink annotations. We explore user reaction to reflow in common cases, how sensitive users are to reflow errors, and how important it is that personal style survive reflow. Our findings can help designers and system builders support freeform annotation more effectively.

Keywords
Digital ink, annotation, context, reflow, handwriting recognition, documents, annotation system design.

1 INTRODUCTION

Free-form document annotation is a crucial part of every knowledge worker's life. Despite the exponential improvement in computer performance, when it comes to reading and annotating documents people still turn to pen and paper. This is reasonable, as pen and paper offer many advantages. One key advantage is the ease with which the reader may sketch unstructured notes and drawings in response to document content.

There are definite advantages to emulating this annotation ability on a computer. While real ink annotations often end up in the recycle bin, digital annotations can persist throughout the lifetime of a document. They can be filtered and organized, and, like digital documents, they can easily be shared.

Now that email and the World Wide Web are well established, the number of digital documents people interact with on a daily basis has increased dramatically. Unlike their paper counterparts, these documents are read in many different formats, and they are displayed on diverse devices and in different-sized windows. They may also be edited, included in other documents, or they may even dynamically adapt their contents. All of this means that any given document may reflow to many different layouts throughout its lifetime.

The lack of a permanent layout poses a unique challenge in the adaptation of freeform pen-and-paper annotation to the digital domain: Each time the content of a digital document reflows to a new layout, any digital ink annotations must also reflow to keep up with it.

This represents a significant technological challenge. In order to meet it, we must follow three broad steps: First, as a user is marking up a document, we must group and classify their ink strokes according to rough annotation categories (e.g. underline, circle, connector, margin comment, etc.). Second, we must anchor each annotation to its surrounding context in the document. And third, when the layout of the underlying document changes, we must transform each annotation to agree with the new layout of its context.

This third and final step is the primary focus of this paper. We have implemented an initial, straightforward approach to the problem of reflowing ink annotations, and there is much work left to do in refining it and developing it into a working solution. Before we develop our approach further, however, there are significant empirical questions we must answer in order to guide our future research.

For instance, what do people expect to happen to their annotations when the underlying document reflows? Does our initial approach achieve the most basic requirement of reflowable annotations, to preserve each annotation's contextual meaning? And do users prefer their own original ink when viewing their own annotations, or are more formalized versions (which are technologically easier to reflow) acceptable? Most people are not familiar with the

[1] Current address: Department of Computer Science, Brown University, 115 Waterman St, Providence, RI 02912

experience of having their ink reflow, and so their reactions are largely unknown.

Many groups have addressed handwriting and diagram recognition issues, some have looked at annotation anchoring, and some have even looked at modifications to existing ink (for instance "prettying" handwriting), but none to our knowledge has addressed the issue of how users react when their free-form digital ink annotations are automatically reflowed.

This paper therefore offers three key contributions. First, after a brief outline of related work, we propose a simple framework for thinking about the issues involved in reflowing digital ink annotations. Second, we report on user reaction to having their annotations reflowed. And third, we outline the important lessons we learned from our early experience implementing and user-testing reflowing digital ink annotations.

2 RELATED WORK

Several groups have worked in areas related to reflowing digital ink annotations, including digital ink recognition, anchoring annotations in digital documents, modifying ink, and the use of free-form digital ink for annotation.

2.1 Digital Ink Recognition and Sketching Interfaces

Most of the effort in digital ink recognition has historically centered on handwriting recognition [14]. Yet while interpreting a handwritten comment may help in selecting a suitable anchor for it in the document text, it is sufficient to know that a set of ink strokes *is* handwriting — without knowing what it says — to do a good job of reflowing it [1]. We currently do no automatic handwriting recognition in our prototype system.

More recently, researchers have looked at ink shape recognition to support a variety of sketch-based interfaces. Landay *et al.* have done extensive research on sketching user interfaces [8, 10, 11]. Their work is partially grounded in Rubine's work on pen gesture recognition [15]. Similarly, Gross and Do have looked at recognizing and parsing sketched architectural diagrams [6]. These systems use heuristics or machine learning techniques to recognize a set of shapes or gestures, and similar techniques could be brought to bear on reflowing annotations. However, these systems do not associate strokes with an independent context such as an underlying document, and so did not need to modify the user's ink to agree with changes in this context.

Golovchinsky and Denoue have experimented with shape recognition for digital ink annotations [5]. They initially tried simple heuristics; however they found these to be insufficient. Instead, they propose using automatic machine learning techniques. Our prototype system currently relies on manual classification of annotations. While this is clearly not a satisfactory working solution, there may be a middle ground where automatic classification is performed when possible, and the user is consulted when it is not.

2.2 Anchoring Annotations

A number of groups have explored the problem of anchoring annotations so they "keep up" with the document when its layout changes or its content is edited.

Phelps and Wilensky [13] proposed a framework for robustly anchoring annotations in a digital document using features extracted from its structure and content. Although their framework is useful, they only dealt with logical anchor position and did not investigate how layout changes would affect the appearance of annotations.

Brush *et al.* [3] did consider the appearance of repositioned annotations in frequently modified digital documents. Similar to our work, they evaluated the effectiveness of their algorithm before real users. Unlike our work, however, users of their system explicitly selected the text to which each annotation was anchored, and they only used formalized text and highlighter annotations rather than freeform ink annotations.

2.3 Modifying Digital Ink

When reflowing ink annotations it may be useful to separate the drawing style of the annotation from its substance, and then reapply the style later on. Several projects have successfully examined this approach for free-form drawings [7, 4], but have so far only considered local features as candidates for style. However, global properties of an ink annotation are an important aspect of its style. For instance, if the kink in the middle of a curly bracket is applied as a local stylistic feature it may be erroneously reproduced several times in the reflowed figure.

Other systems have attempted to recognize and convert users' ink strokes to formalized structures [9, 18]. This is similar to the "clean" version of annotations that our system produces. Indeed, Arvo [2] and Zanibbi [17] have done work on automatically cleaning up user ink strokes after they have been interpreted. Although their results are very relevant to reflowing ink annotations, their work did not deal with adaptation to an underlying context.

2.4 Free-form Digital Ink Annotations

Golovchinsky and Schilit [5, 16] describe XLibris, an active reading system that allows for freeform digital ink annotation of documents. The system reflows annotations using a stretching and splitting approach that is similar to the style-preserving reflow technique in our software. While XLibris was used to study novel uses of digital annotations, no formal studies were performed on user reactions to reflowed annotations or their expectations of such a system.

3 FRAMEWORK

Now we turn to a description of our framework for support of reflowing digital ink annotations. Broadly speaking, enabling reflowable annotations requires that we be able to satisfy the following three criteria:

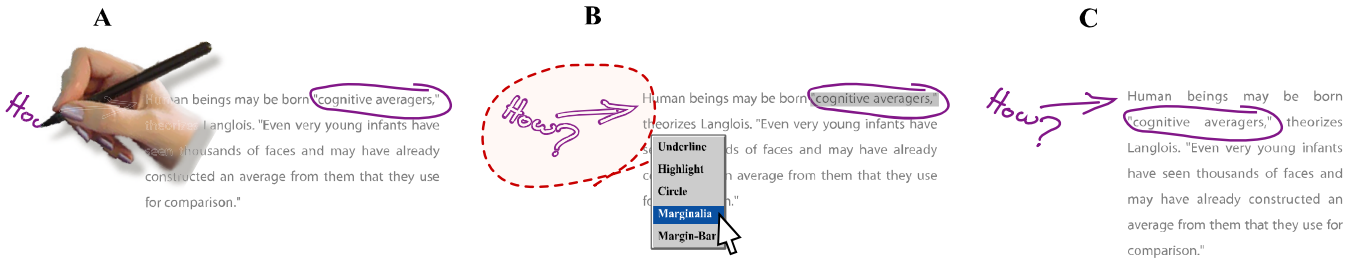

Figure 1: Annotation Reflow Framework. (A) A user annotates the document. (B) Strokes are grouped and classified as annotations and anchored to the text. Manual grouping and classification is shown. The circle's anchor is highlighted). (C) The system reflows the annotations to agree with changing document layout.

3.1 Grouping and Classifying Ink Strokes

First, as the user writes or draws on a document (**Figure 1A**), we must be able to group and classify individual ink strokes into annotations, at roughly the same level of abstraction and with the same accuracy as a human would (**Figure 1B**). Ideally, this would be done with the least possible interference in the primary annotation task. Thus, an automatic approach may be preferable to one directed by the user.

There are several choices for when automatic grouping and classification should occur. If they occur after each ink stroke is created, the system may appear more responsive when the document is reflowed, however it may also make it less responsive while the author is creating annotations. On the other hand, if they are only performed when the document is reflowed (that is, only when the information is needed), it will have the opposite effect.

Grouping ink strokes may be based on the temporal order of strokes, the spatial arrangement, or a combination of both. Relying on temporal order is fast and easy, and it exploits the fact that the strokes comprising most annotations are created sequentially. However, there are times when strokes are created out of order (for instance, when the user goes back to dot an i or cross a t), in which case spatial arrangement may be a better criteria for grouping.

Finally, ink annotations are ambiguous by nature, and improper classification can yield confusing behavior. For example, if a horizontal arrow pointing into the text is erroneously classified as an underline, the arrow might get split across multiple lines when reflow occurs. Thus, totally automatic grouping and classification may not be feasible. Instead, a hybrid approach may be taken, such as requesting the user's help when automatic grouping and classification yield a low confidence decision. This approach has its dangers, though, as anything that interferes with the user's primary task of annotating the document may be confusing or annoying. See **Figure 1B** for an example of a manual classification user interface.

3.2 Anchoring Annotations

After ink strokes are grouped, we must be able to anchor each classified annotation to its intended context (i.e., where it belongs in the document) such that the context can be recovered even if the document's layout, format, or content changes (**Figure 1B**).

Real pen-on-paper annotations are affixed to a particular position on a page. However, physical position in a digital document loses meaning when the document reflows. Instead, the annotation must be anchored to its surrounding logical context (for instance, the range of text it is near).

This is challenging for two reasons: First, digital ink annotations often do not offer a strong indication of what they should be anchored to. Comments in the margin, for instance, may pertain to a text range in the immediate vicinity, or on the other side of the page. Second, the document may be edited between when it was originally annotated and when the annotation is displayed, making it harder to recover the logical context.

An effective anchoring scheme must therefore accommodate these possibilities. One way to achieve this is to employ sophisticated automatic analysis of document contents to determine a "robust" anchor, such as in [3]. This approach can still result in errors, though. Another possibility is to ask the user to explicitly specify what the anchor is for a given annotation. As in grouping and classification, though, this approach may distract the user from their primary annotation task.

3.3 Rerendering Annotations

Finally, after an annotation has been anchored to a document, we must be able to modify it in such as way that if the document layout changes, the annotation naturally "keeps up" with its context, yet it retains its visual meaning and style (**Figure 1C**).

A user of an annotation system may expect reflowed annotations to be visually similar to the originals. However, redrawing an annotation so that it is as similar as possible to the original is both an ill defined problem and technically difficult. We consider two possible approaches. The first is to manipulate the user's original ink so that it fits the new context. However, this manipulation may introduce visual artifacts and is difficult to perform on many annotation types (what is the best way to break a circle over two lines?). A second approach is to draw a stylized version of the interpreted annotation. While this is significantly easier to do, it may prove disorienting to the user. Moreover, the system's rendering style may not suit the tastes of its users.

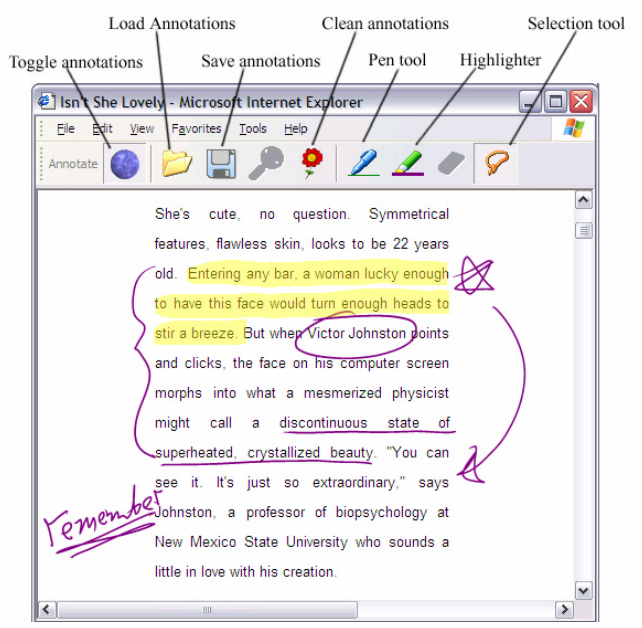

Figure 2: The Callisto toolbar is an IE plug-in that provides facilities for drawing and classifying annotations on any web page.

4 STUDY

We conducted a user study to gauge people's reaction to having their annotations reflowed by software written to conform to our framework. We wanted to examine what people expect of annotations when a document reflows, and under what conditions reflowed annotations are effective. We determined to test three main hypotheses:

First, we expected that users would want their annotations to reflow with document content. This is a key question for our research. Perhaps, for instance, the ability to view a document at different aspect ratios is not as important as preserving annotation placement.

Second, we expected reflowed digital ink annotations to preserve their original context and interpretation. Any annotation reflow strategy must meet this criterion. The interpretation of most annotations is highly dependent on their context. The meaning of a star drawn beside a particular sentence, for example, changes dramatically if it moves down a line or two. We expected our prototype system to be moderately successful in preserving context, and wished to study under what conditions it fails to preserve it.

Finally, we expected that people would occasionally prefer "cleaned up" versions of their annotations. For instance, they may prefer that their annotations be cleaned up before sharing them in collaborative scenarios, or in compensation for the difficulty of writing with a stylus. Some researchers have noted that automatic layout changes may be disorienting to users [17], and that maintaining an informal rendering style denotes transience and encourages creativity [9], so any large response in favor of cleaned up annotations could be seen as a surprise.

4.1 Software

We conducted our study using the Callisto digital ink annotation plug-in for Microsoft Internet Explorer (**Figure 2**). We designed and implemented Callisto based on our framework. It supports an IE toolbar with a pen and a highlighter tool that allow a user to mark any part of any web page with digital ink. Ink strokes persist in a local cache on the user's machine.

To group and classify raw ink strokes into annotations, the user manually selects a set of strokes with the Callisto selection tool, and then chooses a classification from a predetermined list of annotation types (illustrated in **Figure 1B**). The list of annotation types includes underlines, highlights, marginalia (e.g. margin comments and symbols), circles, and margin bars. Choosing one of these categories automatically anchors the set of strokes as an annotation to a context in the document, using a set of simple heuristics (see **Figure 3A**). Even though Callisto supports automatic ink stroke grouping and classification, we chose to use Callisto's manual classification interface in our study in order to marginalize the effects of automated classification errors in our results.

4.1.1 Rerendering Annotations

After classification, whenever the IE document window is resized, Callisto automatically rerenders any annotations on the current page so that they keep up with their contexts (See **Figure 3B**).

When redrawing an underline annotation, we retrieve the line of text each underline ink stroke is associated with. Wherever a line of text breaks into two lines, we split the corresponding stroke. Wherever two consecutive lines of text become one, we join the two strokes and apply a low-pass filter to the new stroke to eliminate any artifacts caused by the join operation.

Since circles are anchored to a single line of text, we simply scale each circle relative to the bounding rectangle of the anchor text. If the anchor text splits over two or more lines, we copy the circle and scale each copy to each section of anchor text. While this technique produces reasonable results for circled words or short sentences, it does not work well for large sloppy circles, or for circled passages or paragraphs.

The content of margin comments varies greatly. We do not modify them, but rather reposition them vertically to align their peak with the top of the anchor text. We position margin bars in the same manner, except we scale them to the new height of the anchor text.

Besides rerendering annotations to preserve the user's original drawing style as in **Figure 3B**, our system can also draw formalized "cleaned-up" versions of annotations (**Figure 3C**). This turns out to be fairly simple for most annotation types, and our system takes a straightforward

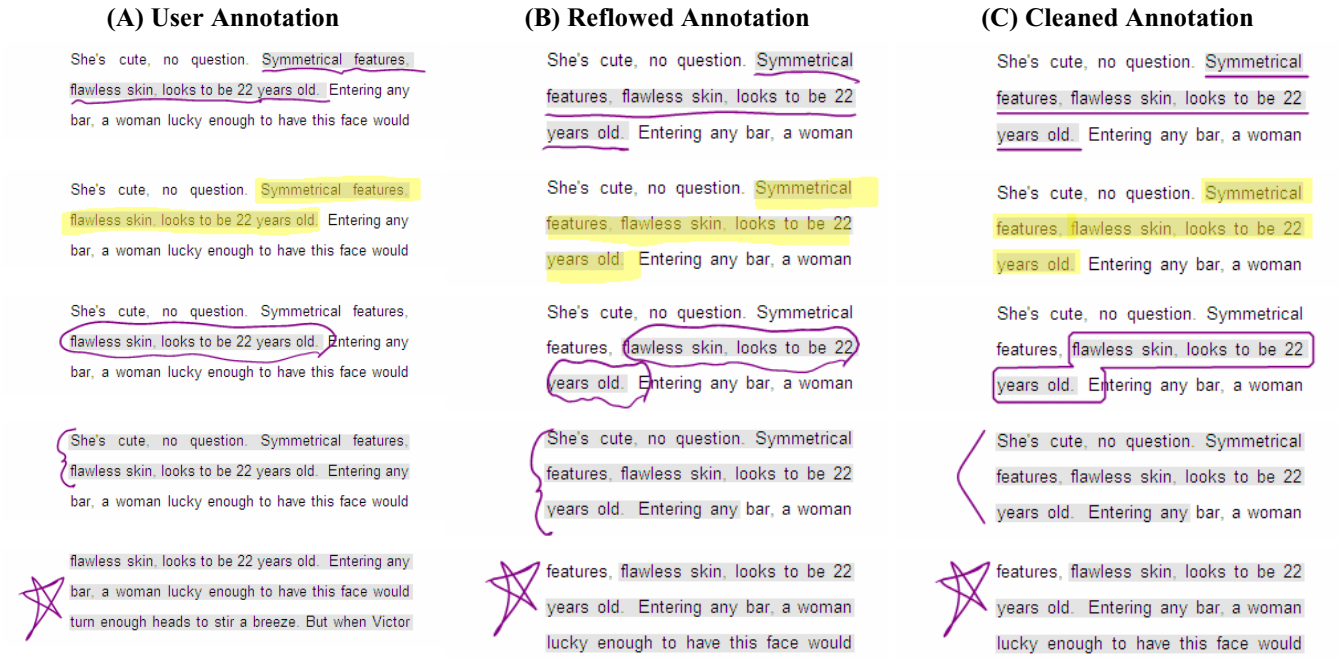

Figure 3: (A) The user's annotations are anchored to neighboring text (shown in gray). (B) The annotations are reflowed so as to maintain their relationship to their anchor text. (C) Annotations may be "cleaned" by rendering them in a stylized manner.

approach: It renders underlines as straight line segments, highlights as translucent horizontal rectangles over the anchor text, circles as round-cornered rectangles, and margin bars as simple Bezier curves spanning the height of the anchor text. Since we do not parse margin comments and symbols, we simply leave them as the user drew them (see the bottom row of **Figure 3).**

4.2 Hardware

During the study, we ran Callisto on an Acer TravelMate 100 running Microsoft's Tablet PC operating system (**Figure 4**). This machine is a laptop computer with a pen input digitizer integrated into its screen. The screen swivels and folds back to cover the keyboard, thus allowing it to mimic the form-factor of a writing pad. Users interact with the Tablet PC with a stylus, which can be used to control the cursor and to draw and write directly on the screen.

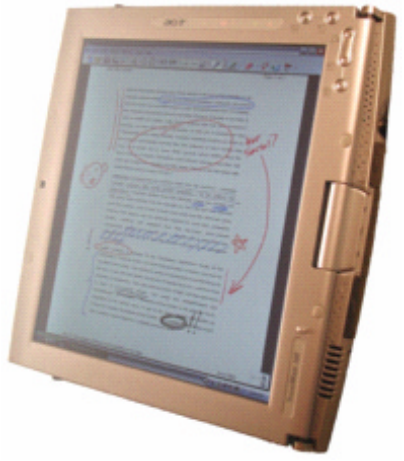

Figure 4: Acer TravelMate 100 running Microsoft Windows XP Tablet PC Edition and Internet Explorer with Callisto plugin.

4.3 Experimental Method

4.3.1 Task

Participants in our study were asked to perform two tasks. The first was to read and annotate an unpaginated 1500 word general interest science article on the Tablet PC in a portrait-mode window. Participants were told they had to give a brief talk about the article to a class or club they belonged to, and to make whatever type of marks they would usually make to help them understand the document and remember their reactions to it.

4.3.2 Conditions

When a participant completed the first task, an experimenter manually grouped and classified the participant's annotations. Then the participant was asked to perform the second task, rating the annotations under a number of reflow conditions.

We designed six experimental conditions to explore user reactions to reflow. The first three conditions—*narrow, wide,* and *edited*—focused on common causes of reflow, namely change in the width of the document window, and modification of the document text. In the *narrow* condition we displayed the document in a window comparable in width to the screen of a PDA (Personal Digital Assistant). In the *wide* condition we switched the Tablet PC to landscape mode to display the article in a window such as might be found on a desktop computer. In the *edited*

condition, the document was displayed as in the *wide* condition; however the article content had been slightly edited.

The next two conditions explored the desirability of "prettying" annotations: In the *cleaned-original* condition, annotations were rendered in a stylized manner as described in section 4.1.1 at the same page width used for the annotation task. In the *cleaned-wide* condition, stylized versions of the annotations were displayed on the document at the same width as in the *wide* condition.

We designed the last condition, *jittered*, to measure how sensitive user perception of different types of annotations is to reflow errors. For this condition we displayed the document in a *wide* sized window, where each annotation was randomly offset from its correct position.

4.3.3 Ratings

For each annotation, participants were asked to rate the following five questions in the *narrow, wide, edited,* and *jittered* conditions (or whatever subset of these conditions they had time to complete):

1. The original intent of the annotation has been preserved.
2. The appearance of this annotation is acceptable.
3. This annotation is noticeably different from the original.
4. Given the current document layout, I would have made a different mark here.
5. It would have been better to "freeze" the document than to redraw this annotation.

Participants responded on a six point Likert scale where a rating of "1" indicated strong disagreement, and a "6" indicated strong agreement. In addition, the following extra question was asked in the *narrow* and *edited* conditions:

6. How important is this annotation to you?

For this question, a response of "1" indicated the annotation was not at all important, while a "6" indicated it was very important.

In the two *cleaned* conditions, participants gave ratings for the same first two questions as in the other conditions, however questions 3 and 4 were replaced by the following (also rated on the 6-point Likert scale for agreement):

3. For my own use, I prefer the cleaned up annotation to the original.
4. For sharing with others, I prefer the cleaned up annotation to the original.

The survey also asked the participants to comment on what they liked and disliked about each annotation in each condition.

4.3.4 Procedure

Participants performed the annotation and rating tasks individually during one and a half hour sessions in a controlled environment. They began the session with a five-minute practice task instructing them in the use of the Tablet PC hardware and the Callisto plug-in software.

Each participant evaluated their first 20 annotations in 2 or 4 of the 6 conditions (depending on how much time he or she spent on each condition). 13 participants evaluated their annotations in the *narrow* and *wide* conditions, 6 evaluated the *edited* and *jittered* conditions, and 9 evaluated the *cleaned-original* and *cleaned-wide* conditions. We counter-balanced the order in which participants encountered the *narrow, wide, edited,* and *jittered* conditions to reduce ordering effects. The *cleaned-original* and *cleaned-wide* conditions were always evaluated last.

4.3.5 Participants

Participants were 18 residents of Redmond, WA and the surrounding area, aged 20 to 50. They all had at least some college education, and all spent a minimum of 30 minutes a day using a computer for tasks such as surfing the web. The participants received software-packages of their choosing for taking part in the study.

4.3.6 Analysis

We analyzed individual participant ratings using SPSS, a standard statistical analysis package. Among other things, we observed that ratings for the first four questions asked about each annotation were highly correlated. Based on this observation, we calculated a single scalar *goodness* score for each annotation in the *narrow*, *wide*, *edited*, and *jittered* conditions. We did this by grouping each set of answers to questions 1 through 4 into a list of vectors (one vector per annotation per condition). We then ran Principal Components Analysis (PCA) on these vectors, and projected each onto the first eigenvector (since the first eigenvalue dominated over all others). This resulted in a single scalar score for each annotation, which was then normalized to fall between 1 and 6. We performed a similar procedure to calculate a *goodness* score for annotations rated in the *cleaned-original* and *cleaned-wide* conditions.

5 RESULTS

The 18 participants in our study made and rated 415 annotations. 36 of these annotations were affected by bugs in the Callisto software, and the ratings for them were excluded from our analysis. **Table 1** shows the remaining 379 annotations grouped by type. The distribution of types in **Table 1** closely resembles the distribution of types made in previous studies [12].

Table 1: Frequency of annotation types created by participants.

Annotation Type	Total	%
underline	118	31%
highlight	102	27%
marginalia	90	24%
circle	44	12%
margin bar	25	6%
Total	379	100%

To discern whether people indeed want their annotations to reflow with the document content, we examined responses to the question of whether it would have been better to freeze the document than to reflow a given annotation. We found a significant correlation between this question and all of the others asked for each annotation. In particular, there is a strong negative correlation with the appearance acceptability question (Pearson $r(529) = -0.698$, $p < 0.01$), indicating that users only prefer the document to be frozen if the appearance of the annotation is not satisfactory.

Essentially, users wanted their annotations to reflow if reflow was done right, and not if it wasn't. This is not a surprising finding, however it is encouraging. It indicates that there is no otherwise unexplained resistance to reflowing ink annotations, so if a system can be built to do it well, people will accept it.

It may also indicate a strategy for automatically choosing when to reflow a document against when to freeze it: If there are known annotations that will fail under reflow, then freeze the document. Unfortunately, running three-way analysis of variance (ANOVA) on each of the ratings questions turned up no significant main effect for annotation type or reflow condition, indicating that there may not be a simple criterion for automatically deciding when to reflow and when not to.

When we look at whether reflowed digital ink annotations preserved their original context and interpretation, we find some interesting data. First, the direct preservation rating question we asked participants to answer did not yield any significant main or interaction effects when we ran a three-way ANOVA on it with annotation type and reflow condition as factors. So we look to other evidence. In post hoc comparisons comparing different conditions against one another for the appearance acceptability question, we found that the *jittered* condition was not significantly different from any of the other *narrow, wide*, or *edited* conditions. Also, the importance question asked in the *narrow* and *wide* conditions was not strongly correlated to any of the other questions asked. This all indicates that when users' original ink strokes are reflowed, precise preservation of context does not matter as much as we had expected, as long as the gross context is preserved.

When we look to whether people occasionally prefer cleaned up versions of their annotations over annotations that preserve their own style, we find that in fact they overwhelmingly do. For this, we examine the *goodness* scores of annotations rated in the *narrow* and *wide* conditions (where the user's original ink strokes appeared, and nothing was done to artificially alter either the ink strokes or the document), and we compare them to the *goodness* scores of annotations in the *cleaned-original* and *cleaned-wide* conditions. Results of this comparison appear in **Table 2**, and they show that annotations in the *cleaned* conditions were generally rated higher relative to each other than those in the non-*cleaned* conditions.

Table 2: Comparison of *goodness* scores in *cleaned* and non-*cleaned* conditions. *Goodness* scores run from 1 to 6.

Conditions	Always 4 or above	Mixed	Always 3 or below
narrow, wide	96	36	81
cleaned -original, *-wide*	127	41	39

6 DISCUSSION

Clearly, there is much work left to be done before ink annotation reflow works well across the wide variety of common cases in which it may occur. However, the preliminary results we have obtained so far are encouraging. We learned many valuable lessons from our user study, and we offer the following key observations.

6.1 Cleaning annotations raises expectations

Many participants in our study seemed to have higher expectations for their "cleaned-up" annotations than for their non-*cleaned* annotations. One participant looked at one of her cleaned-up underlines and told us that "[the system] should have underlined the whole sentence for me." Another participant told us that his "underlines should have been turned into bold text". This is all the more intriguing when contrasted with the fact that fine-grained context did not matter as much as expected to participants when they considered how their original ink strokes were reflowed in the non-*cleaned* conditions. Apparently, participants took cleaned-up annotations to indicate that the system had understood their intentions (and therefore such annotations should be handled more "intelligently"), whereas they were more forgiving of reflow mistakes when annotations were not cleaned-up.

6.2 There are many types of free-form annotations

Of all annotations users made during our study, 24% were manually classified as marginalia. Hidden in this figure are a number of annotation types that our system does not currently support. Some of these are simple annotations that we had not considered, such as inline marks (quotes, brackets, etc.), and inline textual comments. Others are annotation types such as connectors, which are more difficult to implement due to possible ambiguities in interpretation and rerendering. Although a free-form digital ink annotation system such as Callisto may cover a majority of annotation types with a fixed list of five or six types, it may also be prohibitively difficult to design a system to handle all possible types of annotations upfront. Instead, this may be an area where machine learning techniques could be employed to gradually learn a user's set of annotations (and reflow procedures for them) over time.

6.3 Anchor identification is often ambiguous

Identifying where an annotation belongs in a document using simple anchoring heuristics — by which the annotation is implicitly anchored to the text that is "closest" to it — is often insufficiently precise, especially when participants considered their "cleaned-up" annotations.

Sometimes, for instance, when the user intended a margin comment annotation to correspond to just a few words in the text, our anchor identification algorithm chose the entire paragraph in which the intended text appeared. More work is needed to explore the nature of implicit anchoring, the model of user expectation underlying implicit anchoring, and how automatic algorithms might accommodate it.

6.4 Users need an unobtrusive feedback UI

Some participants in our study were surprised by how Callisto reflowed their annotations. One participant asked "why did [Callisto] put my underline over this picture?" Another commented "this [margin comment] does not belong here, it belongs beside the paragraph just above."

Users should be able to understand how the system has interpreted their annotations — and the ambiguities the system faces in reflowing their annotations — *before* reflow occurs, since afterwards it is too late. They should also be given the opportunity to fix reflow mistakes when they occur. Yet feedback of this form must also avoid interrupting the flow of the user's primary annotation task.

Several projects have attempted to provide such feedback [2, 17], but several questions remain open for study: What is a user's tolerance for making corrections? For example, many people disable the as-you-type spelling checkers built into some word processors, finding them disruptive enough to warrant the risk of sending out an un-spellchecked document. Also, since annotations are generally informal, it seems unlikely that users will be willing to run a batch-mode classification error checker as they sometimes do with traditional spelling checkers. Will users risk improperly anchored annotations rather then spend time fixing inaccurate classifications? Can software feedback be made unobtrusive enough to avoid distracting the user, yet clear enough to avoid surprising the user?

7 CONCLUDING REMARKS

Reflowing digital ink annotations is an important advance over other digital annotation technologies. Though there is still much work to be done, we have described a flexible framework for handling reflowable ink annotations, and the results of our study indicate that this framework is a valid starting point for further research. We hope our research can help establish principals for support of natural, free-form ink annotation on every digital document!

8 ACKNOWLEDGEMENTS

Thanks to Libby Hanna, Sashi Raghupathy, Michael Shilman, P. Anandan, and Jim Kajiya for their invaluable assistance in conducting this research.

9 REFERENCES

1. *aha! InkWriter Handbook.* Microsoft Corp, aha! Software, Mountain View, CA, 1993
2. Arvo, J., and K. Novins. Fluid sketches: Continuous recognition and morphing of simple hand-drawn shapes. In *Proceedings of UIST 2000.* p. 73-80. ACM Press, 2000.
3. Brush, A.J., Bargeron, D., Gupta, A., and Cadiz, J.J., Robust Annotation Positioning in Digital Documents. In *Proceedings of CHI 2001*. p. 285-292. ACM Press, 2001.
4. Finkelstein A., and Salesin, D., Multiresolution Curves, *Computer Graphics (SIGGRAPH '94 Proceedings),* 28(3), p. 261-268.
5. Golovchinsky, G. L. Denoue. Moving Markup: Repositioning Freeform Annotations. In *Proceedings of UIST 2002.* ACM Press, 2002.
6. Gross, M.D., and Do, E.Y., "Ambiguous Intentions: A Paper-like Interface for Creative Design. In *Proceedings of UIST 1996*. p. 183-192. ACM Press, 1996.
7. Hertzmann, A., N. Oliver, B. Curless, and S.M. Seitz. Curve Analogies. *13th Eurographics Workshop on Rendering*, Pisa, Italy, June 26-28, 2002.
8. Hong, J.I. and J.A. Landay, SATIN: A Toolkit for Informal Ink-based Applications. *CHI Letters: Proceedings of User Interfaces and Software Technology: UIST 2000*. 2(2):p. 63-72. ACM Press, 2000.
9. Igarashi, T., S. Matsuoka, and H. Tanaka. Teddy: A sketching interface for 3D freeform design. In *SIGGRAPH 99 Conference Proceedings,* p. 409-416. ACM SIGGRAPH, August 1999.
10. Landay, J.A. and B.A. Myers, Sketching Interfaces: Toward More Human Interface Design. *IEEE Computer,* 2001. 34(3): p. 56-64.
11. Lin, J., Newman, M.W., Hong, J.I., and Landay, J.A, DENIM: Finding a Tighter Fit Between Tools and Practice for Web Site Design. *CHI Letters:* Human Factors in Computing Systems: CHI '99, 1999, p. 576-583.
12. Marshall, C.C., and Brush, A.J. From Personal to Shared Annotations. In *Proceedings of CHI 2002*. p. 812-813. ACM Press, 2002.
13. Phelps, T.A., and Wilensky, R. Robust Intra-document Locations. In *Proceedings of WWW9 World Wide Web Conference*. Amsterdam, The Netherlands, May 2000.
14. Plamondon, R., and S.N. Srihari. On-Line and Off-Line Handwriting Recognition: A Comprehensive Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence.* 22(1), January 2000, p. 63-84.
15. Rubine, D. Specifying Gestures by Example, *Computer Graphics*, 25(4), 1991, p. 329-337
16. Schilit, B.N., G. Golovchinsky, and M.N. Price. Beyond paper: supporting active reading with free-form digital ink annotations. In *Proceedings of CHI '98*. p. 249-256. ACM Press, 1998.
17. Zanibbi, R. K. Novins, J. Arvo, and K. Zanibbi. Aiding Manipulation of Handwritten Mathematical Expressions through Style-Prserving Morphs. In *Proceedings of Graphics Interface*. P. 127-134. 2001.
18. Zeleznik, R.C., and K.P. Herndon, and J.F. Hughes. SKETCH: An interface for sketching 3D scenes. In *SIGGRAPH 96 Conference Proceedings,* p. 163-170. ACM SIGGRAPH, August 1996.